# Vue高级面试题汇总

## 说说你对SPA单页面的理解，它的优缺点分别是什么？

区别于传统的多页面，它只加载一个页面到浏览器。第一次加载时，服务器会返回一个index.html页面，并所需的js，css等资源。当切换页面时，页面不会重新加载，而是通过url的变化，加载异步js，css资源。一般单页面url变化有两种，一种是通过页面hash，别一种是通过页面history。

优点：当好的交互体验，用户不用重新加载页面，获取数据是通异步加载资源的方式，页面显示流畅。

缺点：对SEO优化不好，所有的内容都在一个页面上动态显示。

首屏加载速度慢，初次加载资源耗时过多。

## SPA单页面的实现方式有哪些？

在hash模式中，在window上监听hashchange事件（地址栏中hash变化触发）驱动界面变化；

优点：浏览器兼容性好

缺点：url链接不美观，在使用a标签进行锚点跳转的时候，会有问题。

在history模式中，在window上监听popstate事件（浏览器的前进或后退按钮的点击触发）驱动界面变化，监听a链接点击事件用history.pushState、history.replaceState方法驱动界面变化；

优点：url链接和传的多页面链接一样，url链接对用户友好。

缺点：需要服务器做配置，访问其他地址的时候，统一重定向到index.html地址，而且需要前端配置404页面。

## 说说你对MVC、MVP、MVVM模式的理解

MVC：

M代表Model，专门来处理、存储数据。

V代表View，专门来处理页面的展示。

C代表Controller专门处理业务逻辑

用户操作View，View发送指令到Control，完成业务逻辑处理后，要求Model处理相应的数据，将处理好的数据发送到View，要求View把这些数据展示给用户。

缺点：然而在MVC模式中,Model、Control、View三者相互依赖，修改起来要兼顾其他两者，还是非常困难。

MVP:

Presenter将View和Model完全隔离开，Presenter和View相互依赖，Presenter和Model相互依赖，View和Model不再相互依赖，使代码耦合降低。

因为让Presenter发送数据到View,让View展示，仍然需要大量的、烦人的代码。那么可不可以让View在Model变化时自动更新。

MVVM:

VM代表ViewModel负责视图显示逻辑和监听视图变化，M代表Model变成处理业务逻辑和数据。

当用户操作View时，ViewModel监听到View的变化，会通知Model中对应的方法进行业务逻辑和数据处理，处理完毕后，ViewModel会监听到自动让View做出相应的更新。这些通通交给ViewModel来实现，大大降低了开发成本。

在Vue项目中。new Vue()就是一个ViewModel，View就是template模板。Model就是Vue的选项如data、methods等。在开发过程我们只关注View怎么展示，Model怎么处理业务逻辑和数据。不要去管处理业务逻辑和数据后怎么让View更新，View上有操作，怎么让Model处理这个操作，这些通通交给ViewModel来实现，大大降低了开发成本。

## 说说你对Object.defineProperty的理解

Object.defineProperty(obj,prop,descriptor)方法会直接在一个对象上定义一个新属性，或者修改一个对象的现有属性， 并返回这个对象。

obj：要在其上定义属性的对象。

prop：要定义或修改的属性的名称。

descriptor：将被定义或修改的属性描述符。

descriptor属性描述符主要有两种形式：数据描述符和存取描述符。描述符必须是这两种形式之一；不能同时是两者。

数据描述符和存取描述符共同拥有

configurable：特性表示对象的属性是否可以被删除，以及除value和writable特性外的其他特性是否可以被修改。默认为false。

enumerable：当该属性的enumerable为true时，该属性才可以在for...in循环和Object.keys()中被枚举。默认为false。

数据描述符

value：该属性对应的值。可以是任何有效的 JavaScript 值（数值，对象，函数等）。默认为undefined。

writable：当且仅当该属性的writable为true时，value才能被赋值运算符改变。默认为false。

存取描述符

get：一个给属性提供 getter的方法，如果没有getter则为undefined。当访问该属性时，该方法会被执行，方法执行时没有参数传入，但是会传入this对象（由于继承关系，这里的this并不一定是定义该属性的对象）。默认为undefined。

set：一个给属性提供 setter的方法，如果没有setter则为undefined。当属性值修改时，触发执行该方法。该方法将接受唯一参数，即该属性新的参数值。默认为undefined。

定义descriptor时，最好先把这些属性都定义清楚，防止被继承和继承时出错。

## 说说你对Proxy的理解

对目标对象的操作之前提供了拦截，对外界的操作进行过滤和修改某些操作的默认行为，可以不直接操作对象本身，而是通过操作对象的代理对象来间接来操作对象。

let proxy = new Proxy(target, handler)

target 是用 Proxy 包装的目标对象（可以是任何类型的对象，包括原生数组，函数，甚至另一个代理）;

handler 一个对象，其属性是当执行一个操作时定义代理的行为的函数，也就是自定义的行为。

handle可以为{}，但是不能为null，否则会报错

let test = new Proxy(obj,{

get : function (target,property) { return property in target ? target[property] : 0 },

set : function (target,property,value) { target[property] = 6; },

has: function (target,prop){ if(prop == 'b'){ target[prop] = 6; } return prop in target; }

})

## Object.defineProperty和Proxy的区别

Object.defineProperty

不能监听到数组length属性的变化；

不能监听对象的添加；

只能劫持对象的属性,因此我们需要对每个对象的每个属性进行遍历。

Proxy

可以监听数组length属性的变化；

可以监听对象的添加；

可代理整个对象，不需要对对象进行遍历，极大提高性能；

多达13种的拦截远超Object.defineProperty只有get和set两种拦截。

## Vue的模板语法用的是哪个web模板引擎的吗？说说你对这模板引擎的理解?

采用的是Mustache的web模板引擎mustache.js